Navier-Stokes,

fluid dynamics

&

image inpainting

Introduction

What is image inpainting?

Image inpainting: process of filling in missing data in a designated region of a still or video image

→ remove objects from scenes

→ retouch damaged paintings, photographs

→ revert deterioration in films/ photographs

Current active area of research is to automate digital techniques for inpainting.

Today's topic:

Introduction of an algorithm for digital inpainting of still images

Method:

- Find analogy between fluid dynamics and transportation of image intensity
- Use Navier-Stokes equations for incompressible fluids
- Idea: user selects a region to be inpainted
  → algorithm transports information into this region

Analogy to fluid dynamics

Regard the Navier-Stokes-equations for an incompressible Newtonian fluid:

$$v_t + v \cdot \nabla v = -\nabla p + \upsilon \Delta v$$
$$\nabla \cdot v = 0,$$

where v is the velocity, p is the pressure and υ is the viscosity of the fluid.

2D-case: ⟶ there exists a stream function Ψ, which satisfies $\nabla^{\perp}\Psi = v$, where

$$\nabla^{\perp} := \left(-\partial_{x_2}, \partial_{x_1}\right)$$

For the vorticity $w := \nabla \times v$, it holds: $$w_t + v \cdot \nabla w = \upsilon \Delta w$$ $$\Delta \Psi = w_3$$

Proof:

- Regard 2D-case in 3D-equations $\Rightarrow v_3 = 0, \dfrac{\partial(\ldots)}{\partial x_3} = 0$
- Apply the operator $\nabla \times (\ldots)$ to the equation system $\Rightarrow$ equations (1), (2) cancel out
- Regard third equation (right hand side):

$$\frac{\partial}{\partial x_1}\left(-\frac{\partial p}{\partial x_2} + \upsilon \frac{\partial^2 v_2}{\partial x_1^2} + \upsilon \frac{\partial^2 v_2}{\partial x_2^2}\right) - \frac{\partial}{\partial x_2}\left(-\frac{\partial p}{\partial x_1} + \upsilon \frac{\partial^2 v_1}{\partial x_1^2} + \upsilon \frac{\partial^2 v_1}{\partial x_2^2}\right)$$

$$= \underbrace{-\frac{\partial p}{\partial x_1 \partial x_2} + \frac{\partial p}{\partial x_2 \partial x_1}}_{=0} + \underbrace{\upsilon\left(\frac{\partial^2}{\partial x_1^2}\left(\frac{\partial v_2}{\partial x_1} - \frac{\partial v_1}{\partial x_2}\right) + \frac{\partial^2}{\partial x_2^2}\left(\frac{\partial v_2}{\partial x_1} - \frac{\partial v_1}{\partial x_2}\right)\right)}_{=\upsilon \Delta w_3}$$

Navier-Stokes, fluid dynamics & image inpainting

- Introduction
- Analogy to fluid dynamics
- Computational examples
- Literature

- Look at left hand side:

$$\frac{\partial}{\partial x_1}\left(\frac{\partial v_2}{\partial t}+v_1\frac{\partial v_2}{\partial x_1}+v_2\frac{\partial v_2}{\partial x_2}\right)-\frac{\partial}{\partial x_2}\left(\frac{\partial v_1}{\partial t}+v_1\frac{\partial v_1}{\partial x_1}+v_2\frac{\partial v_1}{\partial x_2}\right)$$

$$=\underbrace{\frac{\partial}{\partial t}\left(\frac{\partial v_2}{\partial x_1}-\frac{\partial v_1}{\partial x_2}\right)}_{=w_{3_t}}+\underbrace{\frac{\partial v_1}{\partial x_1}\frac{\partial v_2}{\partial x_1}+\frac{\partial v_2}{\partial x_1}\frac{\partial v_2}{\partial x_2}-\frac{\partial v_1}{\partial x_2}\frac{\partial v_1}{\partial x_1}-\frac{\partial v_2}{\partial x_2}\frac{\partial v_1}{\partial x_2}}_{=0}$$

$$\underbrace{+v_1\frac{\partial^2 v_2}{\partial x_1^2}+v_2\frac{\partial^2 v_2}{\partial x_1\partial x_2}-v_1\frac{\partial^2 v_1}{\partial x_1\partial x_2}-v_2\frac{\partial^2 v_1}{\partial x_2^2}}$$

$$=v_1\frac{\partial}{\partial x_1}\left(\frac{\partial v_2}{\partial x_1}-\frac{\partial v_1}{\partial x_2}\right)+v_2\frac{\partial}{\partial x_2}\left(\frac{\partial v_2}{\partial x_1}-\frac{\partial v_1}{\partial x_2}\right)=v\cdot\nabla w_3$$

- Look at $\Delta\Psi$: $\Delta\Psi=\frac{\partial^2\Psi}{\partial x_1^2}+\frac{\partial^2\Psi}{\partial x_2^2}=\frac{\partial v_2}{\partial x_1}-\frac{\partial v_1}{\partial x_2}=w_3$ □

Navier-Stokes, fluid dynamics & image inpainting

• Introduction
• Analogy to fluid dynamics
• Computational examples
• Literature

How can we compare 2D-fluid dynamics and image inpainting?

Ψ **stream function** for 2D-fluids $\Leftrightarrow$ I steady-state **image intensity**

Because:

Proposition of an algorithm which projects the gradient of the smoothness of the image intensity in the direction of the isophotes (see [1]) and results in the approximation of the following PDE including anisotropic diffusion:

$$I_t = \nabla^{\perp} I \cdot \nabla \Delta I + \upsilon \nabla \cdot \left(g\left(\left| \nabla I \right| \right) \nabla I \right)$$

For no diffusion and a steady-state solution:

$$\nabla^{\perp} I \cdot \nabla \Delta I = 0,$$

which is the same equation as the vorticity transport equation for a 2D-steady-state inviscid flow.

Conclusion:

Navier-Stokes	Image inpainting
stream function Ψ	image intensity I
fluid velocity $v = \nabla^{\perp}\Psi$	isophote direction $\nabla^{\perp} I$
vorticity $w_3 = \Delta\Psi$	smoothness $w_3 = \Delta I$
fluid viscosity υ	anisotropic diffusion υ

So, we obtain our task:

Instead of solving the transport equation for I, solve the vorticity transport equation for w with anisotropic diffusion:

$$w_t + v \cdot \nabla w = \upsilon \nabla \cdot \left(g\left(\left| \nabla w \right| \right) \nabla w \right)$$

After that, solve the Poisson-equation to get the image intensity:

$$\Delta I = w_3, \qquad I\,|_{\partial\Omega} = I_0$$

Navier-Stokes, fluid dynamics & image inpainting

• Introduction
• Analogy to fluid dynamics
• Computational examples
• Literature

Boundary conditions of our problem: use **Dirichlet-conditions**

Poisson-equation: $\Delta I = w_3, \qquad I|_{\partial\Omega} = I_0$, where I_0 are the pixels on the boundary of the inpainted region

Vorticity-transp.-equ.: In the case of flow:
Need of first or second order schemes to numerically compute $\Delta\Psi$ on the boundary

For images:
Much easier; use some more pixels outside the region to be inpainted to construct accurate boundary conditions for $w_3 = \Delta I$

Advantages:

• No need for specification of special regions where information for the inpainting-action comes from
• Simultaneously fill-ins are possible
• Only thing that has to be done: region that should be inpainted has to be marked

Furthermore:

Inheritation of mathematical theory which is already developed for Navier-Stokes equations

→ Well-posedness of our problem

→ Efficient numerical methods

Computational examples

Example ‘inpaint.cpp’:

- use equation $w_t + v \cdot \nabla w = \upsilon \Delta w \longrightarrow g = 1$
- use data outside of region to compute boundary vorticity
- solve by forward Euler time stepping and centered differences in space (+ interpolation scheme)
- after one time step: solve Poisson-equation and start from the updated image intensity I again
- use $dx = dy = dt = 1$
- 50 steps for Gauss-Seidel-iteration
- 300 iteration steps for whole iteration, where we use 10 steps with $\upsilon = 0.0$ and after that 5 steps with $\upsilon = 2.0$

Example “stripes”

destroyed image

Navier-Stokes solver →

reconstructed image

Example "stripes"

destroyed image — Navier-Stokes solver → reconstructed image

Navier-Stokes, fluid dynamics & image inpainting

• Introduction
• Analogy to fluid dynamics
• Computational examples
• Literature

Example “fine stripes”

destroyed image → Navier-Stokes solver → reconstructed image

Example "fine stripes"

destroyed image → Navier-Stokes solver → reconstructed image

Example "smooth the picture"

good image — Navier-Stokes solver → destroyed image

Example “smooth the picture”

good image

Navier-Stokes solver →

destroyed image

Example “Ferien-Akademie 2006”

destroyed image — Navier-Stokes solver → reconstructed image

Example "Ferien-Akademie 2006"

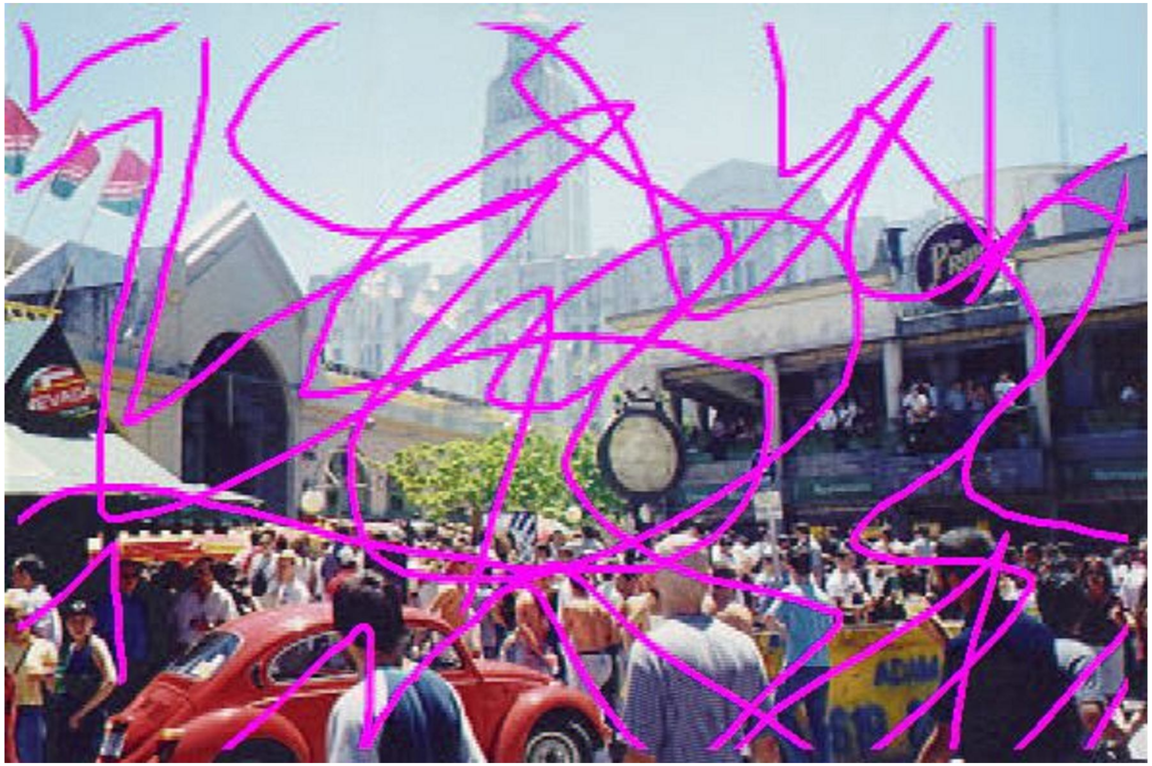
destroyed image → Navier-Stokes solver → reconstructed image

Example from [2]:

• use equation $w_t + v \cdot \nabla w = \upsilon \nabla \cdot \left(g\left(|\nabla w| \right) \nabla w \right)$

• use data outside of region to compute boundary vorticity
• solve by forward Euler time stepping and centered differences in space
• after one time step: solve Poisson-equation and start from the updated image intensity I again
• use $dx = dy = 1, \upsilon = 2$, 50 steps for Jacobi, 5 steps of anisotropic diffusion of I every 10 cycles, 300 iteration steps for whole iteration

Inpainting of stills:



destroyed image → Navier-Stokes solver → reconstructed image

Inpainting of stills:

destroyed image → Navier-Stokes solver → reconstructed image

Do you want some more **nice pictures?**

Alright!

Inpainting by diffusion equations

Problem: “Random” noise

→ Navier-Stokes equations do not help

Idea: Use the nonlinear diffusion equation

$$\nabla \cdot g(u)\nabla u = 0 \qquad \text{in } \Omega_g$$

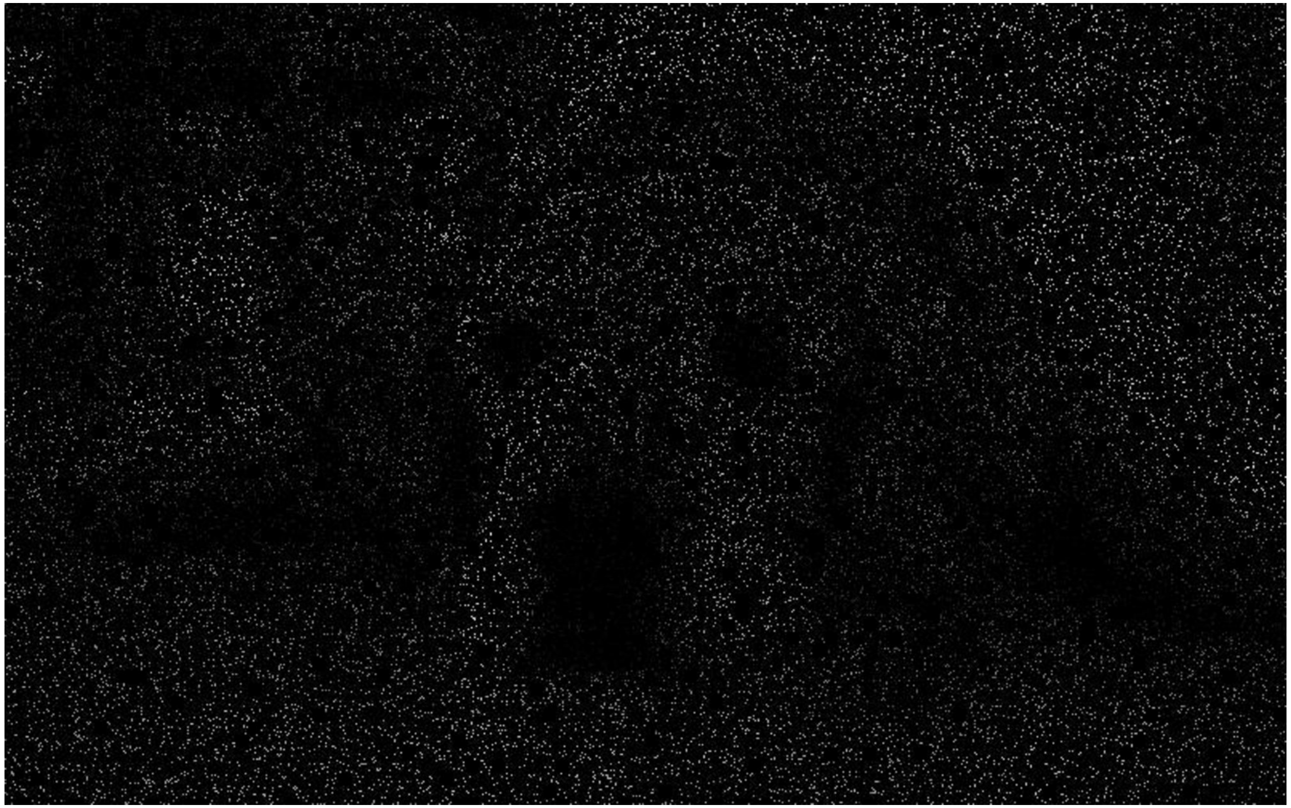
$$u = f \qquad \text{in } \Omega_f$$

$$\frac{\partial}{\partial n} u = 0 \qquad \text{on } \partial\Omega,$$

where Ω_g denotes the noisy and Ω_f the non-noisy part of the picture $\Omega = \Omega_g \cup \Omega_f$.
f denotes the given non-noisy pixels,

$$g(u) = \frac{1}{\sqrt{1+\lambda|\nabla u|^2}}, \lambda > 0 \text{ controls the flow.}$$

Example 1: 95% of random destruction



destroyed image

Diffusion solver →

reconstructed image

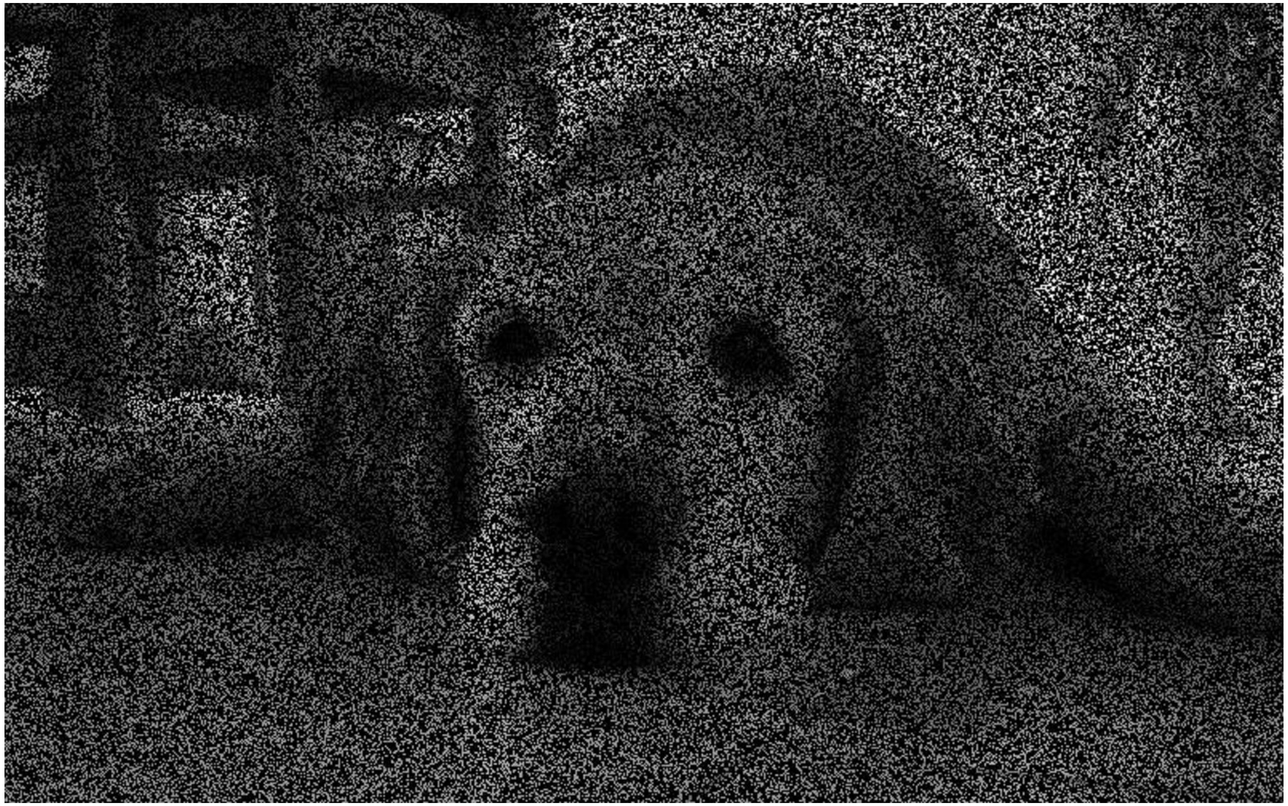
Navier-Stokes, fluid dynamics & image inpainting

- Introduction
- Analogy to fluid dynamics
- Computational examples
- Inpainting by diffusion equations
- Literature

Example 1: 95% of random destruction

destroyed image → Diffusion solver → reconstructed image

Example 2: 70% of random destruction



destroyed image → Diffusion solver → reconstructed image

Example 2: 70% of random destruction

destroyed image → Diffusion solver → reconstructed image

Literature

[1] M.Bertalmio, G. Sapiro, C. Ballester and V. Caselles, Computer Graphics

[2] M. Bertalmio, A.L. Bertozzi, G. Sapiro, Navier-Stokes, Fluid Dynamics and Image and Video Inpainting

[3] F. Durst, Einführung in die Strömungsmechanik

Navier-Stokes,
fluid dynamics
&
image
inpainting

- Introduction
- Analogy to fluid dynamics
- Computational examples
- Literature

That's it!

Thanks for your attention

&

Have a nice day!